THE UNIVERSITY OF NEW MEXICO

ECE 538
Advanced Computer Architecture

Instructor: Lei Yang

Department of Electrical and Computer Engineering

November 1, 2021

QUIZ-3

- Reordering Code to Avoid Pipeline Stalls

Consider the following code segment in C:

```
a = b + e; c = b + f;
```

Here is the generated MIPS code for this segment, assuming all variables are in memory and are addressable as offsets from $t0:

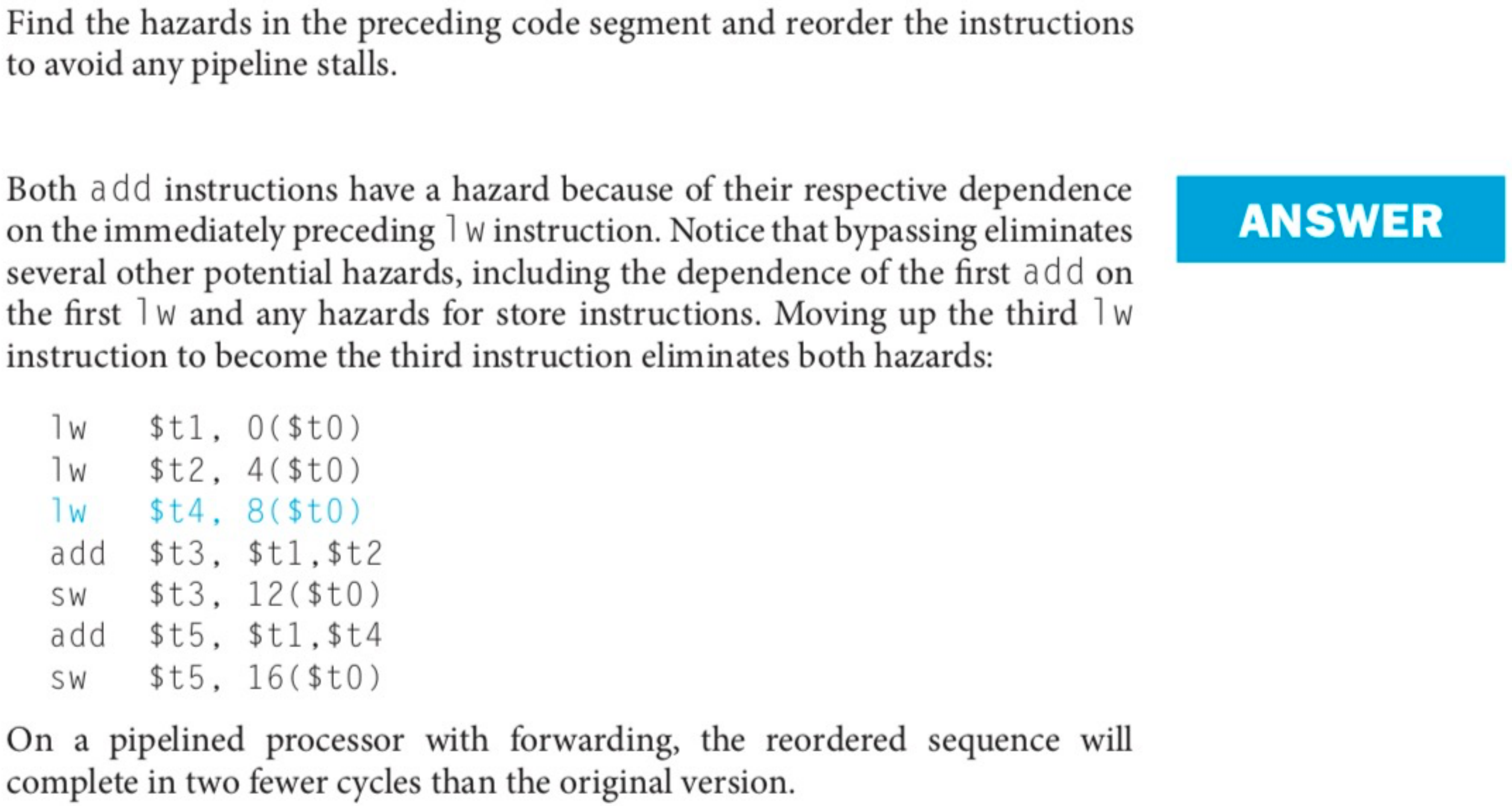
```
lw     $t1, 0($t0)
lw     $t2, 4($t0)
add    $t3, $t1,$t2
sw     $t3, 12($t0)
lw     $t4, 8($t0)
add    $t5, $t1,$t4
sw     $t5, 16($t0)
```

Find hazards in the preceding code segment and reorder instructions to avoid any pipeline stalls.

Answer

- More detailed description will be better in your answers
- Refer to the textbook: Page 280 (Example)

QUIZ-3



Find the hazards in the preceding code segment and reorder the instructions to avoid any pipeline stalls.

Both add instructions have a hazard because of their respective dependence on the immediately preceding lw instruction. Notice that bypassing eliminates several other potential hazards, including the dependence of the first add on the first lw and any hazards for store instructions. Moving up the third lw instruction to become the third instruction eliminates both hazards:

ANSWER

```
lw   $t1, 0($t0)
lw   $t2, 4($t0)
lw   $t4, 8($t0)
add  $t3, $t1,$t2
sw   $t3, 12($t0)
add  $t5, $t1,$t4
sw   $t5, 16($t0)
```

On a pipelined processor with forwarding, the reordered sequence will complete in two fewer cycles than the original version.